UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/900,123	07/05/2001	Scott Wiltamuth	MSFT-0573/160076.1	5765

41505 7590 08/15/2008

WOODCOCK WASHBURN LLP (MICROSOFT CORPORATION)
CIRA CENTRE, 12TH FLOOR
2929 ARCH STREET
PHILADELPHIA, PA 19104-2891

EXAMINER	
VO, TED T	

ART UNIT	PAPER NUMBER
2191	

MAIL DATE	DELIVERY MODE
08/15/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	09/900,123	WILTAMUTH ET AL.	
	Examiner	Art Unit	
	TED T. VO	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *29 April 2008*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-3,6-15,17-19,23-26,28-37,39-41,58-63 and 66-75* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-3,6-15,17-19,23-26,28-37,39-41,58-63 and 66-75* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the communication filed on 04/29/2008.

Claims 1-3, 6-15, 17-19, 23-26, 28-37, 39-41, 58-63, 66-75 are pending in the application.

Response to Arguments

2. All Applicants' arguments have been considered. However, the arguments are not persuasive. The present application as it is noted in the title: Implementing an Explicit Interface member in a computer programming language. Thus, the specification is merely adding code in a program. Furthermore, the claims also appear reciting adding code in a program:

> *A method for operating a computer using object-based computer code of an object-oriented programming language, the method comprising:*
> *utilizing an explicit interface member mechanism* (note #a: programming element) *that enables a class to implement an explicit interface member by explicitly specifying the relationship between the class and the explicit interface member* (note #b: programmed with programming elements)*, wherein the explicit interface member mechanism enables an implemented explicit interface member to be excluded from a public interface of said class; and*
> *storing said class in a form that includes said implemented explicit interface member in a computer readable storage medium* (note #c: i.e. save a program having programming elements in a file/folder/directory of a computer storage like a hard disk, etc.) .

By the identification of notes #a and #b, the acts of claiming are merely the manipulations like writing code using a predetermined code resources such a member of C#, "*explicit interface member*". By the context in the claim note #c, it appears saving a program in a directory.

The claim as a whole fails to implement a practical result because it is as whole includes mere acts of #a, #b and #c. It should be noted that the acts of #a, #b and #c are only programming or a setting a structure of a program. Linking the program to *computer readable storage medium* acts like storing a program in a computer, and produces no real word result.

Thus, claims are interpreted based on the functionality of the claims as a whole, "writing code", structuring code of a particular programming language, i.e. it is a mere common knowledge done by a programmer.

In the C#: 1. For purposes of implementing interfaces, a class or struct may declare explicit interface member implementations. 2. An explicit interface member implementation is a method, property, event, or indexer declaration that references a fully qualified interface member name.

For example

```
interface ICloneable
{
  object Clone();
}
interface IComparable
{
  int CompareTo(object other);
}
class ListEntry: ICloneable, IComparable
{
  object ICloneable.Clone() {...}
  int IComparable.CompareTo(object other) {...}
}
```

Here, ICloneable.Clone and IComparable.CompareTo are explicit interface member implementations, which are used in the claims.

Thus, the claims merely only recited programming per se, as it is clearly claiming the type of above program. Even being associated with a method or storage medium, the claims fail to produce any real world results. Association with method (writing that program)) or storage medium (storing that program) remains programming instructions and its syntax per se. The claims fail to meet claimed statutory under 35 USC 101 because a program per se will not produce any practical results.

Claim Rejections - 35 USC § 101

3. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

4. The claims 1-3, 6-15, 17-19, 23-26, 28-37, 39-41, 58-63, 66-75 are rejected under 35 U.S.C 101 because the claimed invention is directed to non-statutory subject matter.

Claims 1-3, 6-15, 17-19, 23-26, 28-37, 39-41, 58-63, 66-75 fail to meet claims statutory because the claim directs only instructions used in a programming. The claims clearly address C# instructions resources and how they structure a program, like one introduced FORTRAN, C, BASIC using a structure of IF then Else statement for meeting syntax/semantic requirement. Such claims, even being connected to a method and storage medium, the result if yielded by the claimed functionality does not cause any effect to the real world because it is only a generic piece of program stored in a computer (See the claim using limitation, storing in a computer

storage medium). Thus, for infringement subject matter of claims, no programmer allows using C# language. Thus, it lacks of "practical application". Therefore, being seen explicitly from the claimed functionality, the claims as a whole is a piece of programming that cannot product a real world result. The claims fail to be statutory under 35 USC 101.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> A person shall be entitled to a patent unless –
>
> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1-3, 6-15, 17-19, 23-26, 28-37, 39-41, 58-63, 66-75 are rejected under 35 U.S.C. 102(a) as being unpatentable over ECMA Standard "Draft C# Language Specification" drafted by multiple corporations, Hewlett-Packard, Intel, Microsoft, in March 2001 (hereinafter:ECMA).

As per claim 1: ECMA teaches

A method for operating a computer using object-based computer code of an object-oriented programming language, the method comprising: utilizing an explicit interface member mechanism that enables a class to implement an explicit interface member by explicitly specifying the relationship between the class and the explicit interface member, wherein the explicit interface member mechanism enables an implemented explicit interface member to be excluded from a public interface of said class; and storing said class in a form that includes said implemented explicit interface member in a computer readable storage medium.

The reference specification provides a user to use C# language specification to program correctly and uses a compiler to generates code from a generated program (see p. 45: "…the compiler produce a warning…", etc). The C# specification provides syntax implementation for programming correctly when defines an explicit interface member mechanism. See sec. 20.4.1: "Explicit interface member implementations". Especially, see a class or struct: "Interface **ICloneable**", an interface Explicit interface member utilizing an explicit interface member mechanism that enables a class (the body of this class) to implement an explicit interface member by explicitly specifying the relationship between the class and the explicit interface member such as:

```
object ICloneable.Clone() {.}
int IComparable.CompareTo(object other) {.}
```

wherein the "Interface **ICloneable**" enables an implemented explicit interface member to be excluded from a public interface of said class.

The reference does not say to store in class or struct in a computer readable medium.

However, this is the work done in for computer and in computer.

Therefore, it is obviously to ordinary in the art at the time to implement the text of this class into a file for being portable, where according to MPEP, make portable cannot be patentable over a reference.

As per claim 2: ECMA further teaches

A method according to claim 1, wherein said specifying of the relationship includes specifying a qualified name of the class., because it is of C# specification properties. See sec. 20.4.1: Explicit interface member implementations.

As per claim 3: ECMA further teaches

A method according to claim 2, wherein said specifying of the qualified name includes specifying an interface name and said at least one interface member name., because it is of C# specification properties. See sec. 20.4.1: Explicit interface member implementations.

As per claim 6: ECMA further teaches ***A method according to claim l, wherein the explicit interface member mechanism enables the class to implement an internal interface not accessible to a consumer of said class.***, because it is of C# specification properties. See sec. 20.4.1: Explicit interface member implementations.

As per claim 7: ECMA further teaches ***A method according to claim l, wherein said explicit interface member mechanism enables disambiguation of a plurality of interface members having the same signature.***

By definition of Signature, the signature of a method consists of the name of the method and the number, modifiers: thus see **IComparable**.CompareTo(object other) {.} and thought out sec. 20.4.1: Explicit interface member implementations.

As per claim 8: ECMA further teaches ***A method according to claim 1, wherein said explicit member mechanism enables disambiguation of a plurality of interface members having the same signature and return type.***

By definition of Signature and C# programming semantic, the signature of a method consists of the name of the method and the number, modifiers: thus see **IComparable**.CompareTo(object other) {.} and thought out sec. 20.4.1: Explicit interface member implementations.

As per claim 9: ECMA further teaches ***A method according to claim 1, wherein in addition to allowing the implementation of public interface members, said explicit interface member mechanism enables the implementation of private interface members.***, because it is of C# specification properties. See sec. 20.4.1: Explicit interface member implementations.

As per claim 10: ECMA further teaches ***A method according to claim 1, wherein said explicit interface member mechanism enables the implementation of a plurality of non-conflicting specific versions of a generic interface*** , because it is of C# specification properties. See sec. 20.4.1: Explicit interface member implementations.

As per claim 11: ECMA further teaches ***A method according to claim 1, wherein the computer code is programmed according to an object-oriented programming language, and said object-***

oriented programming language is one of C#, Fortran, Pascal, Visual Basic, C, C++ and Java. (This limitation is itself to admit it is claiming a programming instruction. Claiming program per se should be subject under 101) The reference is about C# specification properties. See sec. 20.4.1: Explicit interface member implementations.

As per claim 12: ECMA further teaches ***A method according to claim 1, wherein an implementation of an explicit interface member is a method, property, event, or indexer declaration that references a fully qualified interface member name.*** The reference is about C# specification properties. See sec. 20.4.1: Explicit interface member implementations.

As per claim 13 ECMA further teaches

A method according to claim 1, wherein the class names an interface in a base class list of the class that contains a member whose fully qualified name, type, and parameter types exactly match those of the implementation of the explicit interface member. The reference is about C# specification properties. See sec. 20.4.1: Explicit interface member implementations.

As per claim 14: ECMA further teaches ***A method according to claim 1, wherein said explicit interface member mechanism includes an interface mapping mechanism that locates implementations of interface members in the class .*** The reference is about C# specification properties. See sec. 20.4.1: Explicit interface member implementations.

As per claim 15: ECMA further teaches ***A method according to claim 14, wherein said interface mapping mechanism locates an implementation for each member of each interface specified***

in a base class list of the class. The reference is about C# specification properties. See sec. 20.4.1: Explicit interface member implementations.

As per claim 17: ECMA further teaches ***A method according to claim 1, wherein it is not possible to override an explicit interface member implementation, but where an explicit interface member implementation calls another virtual method, derived classes are capable of overriding the implementation.*** The reference is about C# specification properties. See sec. 20.4.1: Explicit interface member implementations.

As per claim 18: ECMA further teaches ***A method according to claim 1, wherein the class inherits an interface implementation is permitted to re-implement the interface by including the interface in the base class list of the software component.*** The reference is about C# specification properties. See sec. 20.4.1: Explicit interface member implementations.

As per claim 19: ECMA further teaches ***A method according to claim 1, wherein said explicit interface member mechanism prevents conflict among specific implementations of a generic interface.*** The reference is about C# specification properties. See sec. 20.4.1: Explicit interface member implementations.

As per claims 23-26, 28-37, 39-41: See related rationale addressed in claim 1-3, 6-15, 17-19.

As per claims 58-63, 66-75: See related rationale addressed in claim 1-3, 6-15, 17-19.

Conclusion

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ted T. Vo whose telephone number is (571) 272-3706. The examiner can normally be reached on 8:00AM to 4:30PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Y. Zhen can be reached on (571) 272-3708.

The facsimile number for the organization where this application or proceeding is assigned is the Central Facsimile number **571-273-8300**.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

TTV
August 08, 2008

/Ted T. Vo/
Primary Examiner, Art Unit 2191